Assignment 1.

Suggestion: Always use "implicit none" as the second line of the program to help the program detect undeclared variables.

```
Program ....
implicit none
...
```

1. Write a program that prints

```
2 to the power of 20 is xxx
```

where xxx is replaced by the value of 2^{20}.

2. Use appropriate loops for programs that calculate the following sums:

a) 1 + .5 + .25 + ...

b) 1 + .5 + .25 + ... +xxx, where xxx>10^{-3}.

c) 55+56+...+100

3. Read an integer variable i and a real variable x from the console repeatedly . Depending on the value of the integer variable, print the following:

integer	action
1	logarithm of x
2	sine of x
3	square of x
"else"	"quit the program"

Use the DO loop without arguments and the CASE statement.

Note: To read variables x and y from the keyboard, type

```
read*,x,y
```

4. Create a file with a pedigree line that contains animal ID, sire ID, dam ID and year of birth:
HOL234HOL001HOL9831998
Write a programs that will read the IDs and year of birth into separate variables.

5. Print the following number:

```
1.0/3.0**(-i),  i=1,10
```

in F, E and G formats using specifications 10.3 and 12.6.

Optional

6. Browse through a list of Fortran functions in your manual. By writing simple programs, determine differences between int, nint, floor and ceiling.

7. Write a program that

- reads an alphanumeric variable of length up to 10,
- for each character of that variable, writes a numeric equivalent of that character.

The numeric equivalent of character c is ichar(c).

Assignment 2.

1. Set matrices A and B to:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 5 \end{bmatrix}, \quad B = \begin{bmatrix} 3 & 1 \\ 4 & 8 \end{bmatrix}$$

Calculate C=A*B as matrix multiplication using loops and matmul. Print. Do results differ?

Write the sum of A as a whole, along dimension 1, and along dimension 2.

Print the first column of A.

Count how many numbers in A+B are larger than 5 and compute their sum; do not use any loop.

2. Initialize matrix x(100,100) to $x_{ij}=100*i+j-1$. Create an internal function

$$tr(n,m) = \sum_{i=n}^{m} x_{ii}$$

and calculate tr(1,10) and tr(90,100).

3. In the following data (to be read from a file), a line containing the dimensions of a 2-dimensional matrix is followed by the data values of this matrix.

```
 2
1.5 3.2
2.7 3.3
4
4 3 2 1
5 4 3 2
6 5 4 3
7 6 5 4
1
10
```

Create the file above. Write a program using the dynamic memory allocation that will:

a) read the dimension
b) allocate a real matrix with the given dimensions
c) read that matrix
d) print that matrix
e) deallocate the matrix
f) continue until the end of file

Optional

4. Create a vector x of 100,000 reals and initialize x(i)=i+1.0/3.0. Store it in files in two formats:
    a) formatted
    b) unformatted

Then read the files and accumulate the sums. Compare timings of either reading using the CPU_TIME subroutine.
Close files with options to delete them.

5. Write to file the following:

```
1 : printed    1 number(s)
1 2 : printed    2 number(s)
...
1 2 3 4 5: printed 10 number(s)
```

without using more than 2 write statements. Use clause ADVANCE='NO'.

Assignment 3.

1. Read descriptions of array functions. Create examples for: maxval, maxloc, pack, reshape, size and unpack.

2. Let A and b be:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 5 \end{bmatrix}, \quad b = \begin{bmatrix} 5 \\ 11 \end{bmatrix}$$

Write a function that calculates the quadratic form: b'Ab. Make this function:

a) internal,
b) module

3. Write a function that calculates a sample from a pseudo-normal distribution using a formula:

$$\sum_{i=1}^{12} u_i - 6, \qquad u_i \sim UN(0,1)$$

Samples from the UN(0,1) distribution are generated by subroutine random_number().

Write a subroutine that, given a vector with samples, calculates mean and SD

Write a program that asks for a number of samples to be generated, generates those samples from a) UN(0,1) and b) N(0,1), and then calculates their mean and SD.

4. Modify function that generates normal samples to accept parameters for mean and variance. Make this parameters optional with default mean=0 and variance=1.

Optional

5. Write a subroutine printnice_r(x,n), that will print a real matrix x(n,n) with nicely aligned columns.

6. Modify function printnice

a) putting it in a module as an internal procedure
b) eliminating the n argument
a) adding an optional argument, which is format for one line

Assignment 4.

1.Write subrouties printnice_int and printnice_real that print real and integer matrices. Overload so that printnice prints both. Test.

2. Write Makefile to compile.

Optional

3. The program contains the data on animals in structure animal. Create a subroutine that given a variable of type animal will print the information on that animal.

```
module animals
type calf
     character (10) :: id
     integer :: year_of_birth
     character:: sex
end type
end module

program registration
use animals
implicit none
type (calf) :: a,b,c

a=calf('small',1997,'M')
! b is also a male born in 1997 but has a different name
b=a
b%id='large'
```

Write subroutine print_am that prints ID, year of birth and sex, with syntax as below:

```
call print_am(a)
call print_am(b)
```

Optional

4. This assignment tests data structures for sparse vectors, operations on data vectors, and operator overloading. All definitions and procedures need to reside in module sparse_vec. Write:
a) data structure for sparse vector
b) subroutine that creates sparse vector from a dense vector
c) subroutine that prints a sparse vector
d) subroutine that multiplies two sparse vectors.

Provide interface to c) with "=" operator
Provide interface to d) with "+" operator

To the last assignment, add interfaces to "+" and "*" that allow to add or multiply sparse vector by a regular vector.

Assignment 5.

1. Run program lsq.f90. This program with related data sets can be found in directory listed on board. Modify the program so that it can read a parameter file containing: name of data file, number of effects and number of levels. The parameter file could contain the following:

```
data_pr1        !name of the data file
2               !number of effects
2 3             !number of levels for each trait
```

2. Apply the modified program to:
   a) the original data
   b) the data as below

management	animal	treatment	record
1	1	1	
	12		
1	2	2	
	14		
1	3	1	
	13		
2	3	1	
	16		
2	2	2	
	18		
3	4	1	
	15		
3	3	2	
	16		
3	2	2	
	19		

Optional

3. Think how the parameter file can be simplified so that the number of effects is not given, i.e., the above parameter file simplifies to:

```
data_pr1        !name of the data file
2 3             !number of levels for each trait; defines the number of traits
```

hint:when the reading is unsuccessful, for example because of too few numbers, the variable associated with iostat becomes ≠0.

4. Examine differences between lsq.f90 and lsqmt.f90. Run the latter. Change R_0 to these

matrices:

$$R_0 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \end{bmatrix}$$

and compare solutions. Should they be different?

Assignment 6.

1. Write a program that creates the inverse of the animal relationship matrix. Use the following pedigree:

animal	sire	dam	
1	3	4	
2	3	4	
3	5		-
4	-	-	

2. Incorporate the program above in lsq.f90. Run it with data of the yesterday's assignment and pedigrees as above assuming var(e)=**I**10 and var(animal)=

a) **I**2.
b) **A**2

Compare solutions.

Hint: Either add contributions within the program or create A separately and add as a matrix

3. Modify parameters in blup.f90 to support the same models as above. Compare solutions. Estimable functions should be identical.

Optional

4. Modify parameters in blup.f90 for a maternal model using parameters from 2b. Possibly compare solutions against another program.
Hint: If the dam is not identified, create a phantom dam.

Very hard

5. In 3, make management autocorrelated with var(management)=5 and ρ=.8. Then, vary the autocorrelation to 0.98 and 0.9 and compare solutions for the management effects.

Assignment 7

DENSEOP

1. This assignment involves module DENSEOP, which is described at http://nce.ads.uga.edu/wiki/doku.php in modules. An example of the use of this module is in directory testdense of a course directory. Because of complexity, the program inside the directory needs to be compiled by the command:

make

For make to work, copy file Makefile as well as all the other files from the same directory. Examine contents of kind.f90 lapack90r.f90 denseop.f90. They are in directory /work/course2014/f90progs/libs.

Consider the following system of equations Ax=b:

$$\begin{bmatrix} 363018 \\ 304123 \\ 182314 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 150 \\ 181 \\ 106 \end{bmatrix}$$

Using dense matrix module DENSEOP and precision (r8):

a) calculate Cholesky decomposition A=LL' and check its correctness by matrix multiplication,

b) calculate eigenvalues and eigenvectors: A=VDV, and verify that VV'=I,

c) calculate the determinant of A

d) solve for x using both symmetric and general solvers; confirm the correctness of solutions by multiplication

e) Calculate the inverse of A and verify that $AA^{-1}=I$.

2. Repeat solving in different data formats: dense and half stored, single and double precision. To create a half stored matrix, convert a full-stored matrix.

Hint: If a regular double precision matrix is defined as

real (r8)::xf(4,4)

a half stored matrix in same precision is declared as

real (r8)::xh(10) !or n(n+1)/2*

Optional - SPARSEM, SPARSEOP

2. The assignments include the sparse matrix module SPARSEM. An example of the use of this module is in file test.f90 in testsparse of the course directory. Because of complexity, the program inside this directory program needs to be compiled by the command:

make

For make to work, copy file Makefile as well as all the other files from the same directory.

For programs with name other than test, make and Makefile can still be used, but replace "test" with the name of your program in file Makefile.

Data structures for different matrix formats are in file sparse.f90 in directory /work/course2014/f90progs/libs/.

Consider the following system of equations Ax=b:

$$\begin{bmatrix} 3 & 6 & 3 \\ 0 & 4 & 1 \\ 2 & 3 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 15 \\ 18 \\ 10 \end{bmatrix}$$

a) Set this system of equation with densem and sparse_hash matrices,
b) convert the format sparse_hash to sparse_ija,
c) print all matrices,
c) solve by iteration,
d) print sparse matrices in internal format (printm(x,'internal')

3. Create a tridiagonal sparse matrix A(1000,1000) in densem format such that:

a(i,i)=10+un(0,1)
a(i,j+1)=un(0.1) for i=1 to 999
a(j,i+1)=un(0.1) for j=1 to 999

a) Generate b(1000) from un(0,1)

b) Solve system of equations Ax=b using densem and sparse_hash formats. Compare timings.

c) Convert A to sparse_ija format and solve Ax=b using FSPAK90

Assignment 8

Create a directory f90 and download file progsf90.tar.gz. This file is available at directory of the course or at nce.ads.uga.edu. Uncompress this file into a newly created directory by:

```
tar xzvf    progsf90.tar.gz
```

Then compile all the libraries and programs:

```
make linux-intel
make all
```

Each program has its own directory and its own Makefile. To recompile any program separately, go to its directory and type

```
make
```

1. Read:
   a) Readme, Installation, and Makefile files in the main directory,
   b) Readme in directory libs. Identify programs that implement reading the parameters, sparse matrix factorization, and hash function.

2. Run blupf90 with examples specified in Appendices A-C. Examples are stored in the directory examples. Parameter files start or end with letters ex.

3. Directory test8 on the course website includes data/pedigree files for a 3 trait maternal model and a parameter file for one trait model. Run blup90 for one trait with the following solving options:
   a) by default iteration (PCG),
   b) by Gauss-Seidel iteration,
   c) by FSPAK.

For details how to change the solving options, see blupf90.f90 and readme.blupf90 in the blupf90 directory. Also look follow links to programs and FAQ at nce.ads.uga.edu

4. For the previous data set, create a parameter file for 3 traits. Run using:
   a) blupf90 with the default solver (PCG)
   b) iteration on-data program bin/blup90iod2 (in course directory)

Optional

3. Modify blupf90.f90 so that it calculates accuracies for the animal effect. Accuracies for an animal in equation i in a single-trait situation is defined as:

$$1-\mathtt{LHS}^{ii}/\sigma_a^2$$

where $\mathtt{LHS}^{ii}$ is the i-th diagonal of the inverse of the left hand side.

Suggestion: Invert by FSPAK and obtain elements of the inverse by function "getm" from module sparsem. Or use "OPTION sol se" for blupf90 - see blupf90 documentation.

Warning: The formulas are correct only without unknown parent groups in the model.

Assignment 9

This assignment involves the blupf90 family of programs. Binaries are available in directory f90progs/bin.

Parameter files for this exercises are in directory example.
Files ending with *99 contain data files for up to 14 type traits. Parameter file exmr99s1 uses these files for a single-trait model, and exmr99s for 3 traits.

1. Browse through FAQ for BLUPF90 programs. Read information on Input files, Frequent mistakes, and REML.

2. Calculate estimates of variance components by remlf90 and airemlf90 using one and three trait models. Record the number of rounds and CPU time. Unix (including Linux) allows for measuring the CPU time by preceding *command* with *time*, e.g.,

```
time remlf90
```

Attention: if computing takes too long, only obtain approximate time per round, or terminate.

3. Read FAQ pages on Gibbs sampling and Post-Gibbs analysis.

4. For one-trait and 3 trait models, obtain 5,000 samples using gibbs2f90. Gave burn-in of 1 and store every sample. Analyze samples with postgibbf90. Find a burn-in and estimate parameters. Re-estimate parameters with burn-in doubled. Are estimates very different?

5. Formulate conclusions. Are they similar to those in the paper “Reliable computing in estimation of variance components”?

6. Directory test8/orig contains original data for the 3 trait maternal model.
   a) examine parameter file renum.par
   b) run renumf90 and examine the printout and files created by renumf90
   c) run blup90iod2
   d) modify renum.par so to consider only 3 generations of pedigrees
   e) run renumf90 and note the number of parents
   f) run blup90iod2 and note the computing time

Optional

7. Convert data in exmr99s to result in one binary trait, one with 3 categories, and one linear. Estimate parameters, first with a linear model (gibbs2.f90) and then with a threshold model (thrgibbs1.f90). Use postgibbsf90 to determine the number of needed samples. Are estimates of heritabilities and genetic correlations between data sets and programs similar?

8. Extend the model to 7 traits and possibly more traits. For additional co-variances, use average of current diagonals for new diagonals, and 1.0 for off-diagonals. How much longer are the computations with 7 traits compared to a single trait model for:

a) AIREMLF90
b) gibbs2f90.